УДК 681.3

М.К. Буза
Белорусский государственный университет
Проспект Независимости, 4, г. Минск, 220030, Беларусь

МЕХАНИЗМ ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ СЖАТИЯ ДАННЫХ

M.K. Bouza
Belorussian State University
Independence Avenue, 4, Minsk, 220030, Belarus

THE MECHANISM OF INCREASING THE EFFICIENCY OF DATA COMPRESSION

В статье исследуется проблема передачи и хранения больших объемов данных. Проводится сравнение различных методов сжатия текстовых данных без потерь. В качестве исследуемых данных выбраны тексты на русском, английском языках и HTML-файлы. Для повышения коэффициента сжатия предложен смешанный механизм сжатия и приведены оценки его работы.

Ключевые слова: большие данные, текстовые файлы, сжатие данных, смешанный алгоритм.

The article examines the problem of transmission and storage of big data. A comparison of different compression techniques for text data off of losses. As research data under study selected texts in Russian and English languages as well as HTML files. To increase the compression ratio of the proposed mixed mechanism and the evaluation of its work.

Key words: big data, text files, data compression, hashing algorithm.

Введение

Функции современных вычислительных систем состоят в приеме, хранении, передаче, обработке информации и выдаче результатов. Сегодня исключительное значение приобретают акции приема и хранения информации, что, в основном, определяется все возрастающим ее объемом. Различные датчики и коммуникационные среды собирают терабайты информации об окружающей среде.

В 2007 году объем выработанной информации превысил возможности технических средств для ее хранения, а информация продолжает интенсивно наращиваться. Информация поступает из интернета, социальных сетей, научных экспериментов и различных девайсов. Аналитики утверждают, что каждую секунду в мире отправляются около 2,9 миллиона электронных сообщений, объем видео, помещаемого на YouTube каждую минуту – 20 часов, объем данных, обрабатываемых Google за день – 24 петабайта (10^{15}), а объем данных переданных/полученных на мобильные устройства – 1,3 эксабайт (10^{18}).

Генерируют информацию бытовые приборы. Активно идет мониторинг в сельском хозяйстве. Так, например, в США результаты мониторинга отдельной особи на среднестатистической молочной ферме в штате Калифорния генерируют около 200 мегабайт информации в год. Значительный объем представляют сведения об имеющейся информации в различных научных отчетах по экономике, других отраслях.

Для обработки таких данных создавались новые инструменты и технологии, начиная с модели MapReduce компании Google и Hadoop от компании Yahoo, ушедшие от прежней жесткой иерархии и однородности данных [1].

В последнее время в мире созданы различные исследовательские центры и институты, которые разрабатывают методы интеллектуального анализа данных (Data Mining), позволяющие обнаруживать в данных ранее неизвестные, нетривиальные знания, которые имеют практическое применение при решении ряда задач. Основные проблемы больших данных обычно связывают с их объемом (volume), скоростью

передачи и обработки (velocity) и неоднородностью данных (variety). Наибольший интерес у исследователей вызывает объем данных.

Появившееся направление Big Data, по анализу и обработке больших данных, не содержит концептуально новых идей по сравнению с имеющимися при обработке данных. Отличие состоит лишь в создании новых подходов к формированию знаний о данных, связанных с успехами в компьютерных науках, в частности, с созданием высокопроизводительных средств (до нескольких десятков петабайт в секунду) и новых информационных технологий [2].

Каждый день население Земли увеличивается на 220 тысяч человек, а к 2020 году жители планеты будут иметь 20 млрд. различных девайсов с выходом в Интернет. В киберпространстве собираются огромные объемы данных о странах, производствах, системах управления и личном пространстве человека. Этот искусственный интеллект уже угрожает существованию интеллекта естественного.

В работе рассматривается задача эффективного хранения информации с возможностью ее последующей обработки. В основу исследований положен механизм сжатия данных [3].

Ниже исследуются различные алгоритмы компрессии информации, проводится их сравнение по ряду параметров, предлагаются, реализуются и оцениваются способы их упорядочения для улучшения значений выбранных параметров.

Анализ алгоритмов

Представление информации любого типа (звук, текст, графика), как правило, избыточно. Удалив такую избыточность, можно существенно уменьшить требования к техническим средствам хранения информации.

Среди алгоритмов компрессии различают обратимые и необратимые, позволяющие либо полностью восстановить сжатую информацию, либо иметь некоторые потери. В одних случаях потери недопустимы, в других - возможны. Степень допустимых потерь (отличие от оригинала) определяется возможностью распакованных данных для дальнейшего использования. Алгоритмы такого рода безусловно обеспечивают более высокую степень сжатия [3,4].

Основной принцип сжатия использует тот факт, что в текстовых файлах некоторые фрагменты частично или полностью повторяются. Используя математическое моделирование, можно определить вероятность повторения определенной комбинации символов. После этого наиболее часто повторяющимся фрагментам присваиваются самые короткие коды, используя различные технические приемы.

Среди множества алгоритмов сжатия без потерь рассмотрим наиболее используемые алгоритмы и их различные комбинации, чтобы достичь наилучших значений выбранных параметров.

Для исследования была выбрана текстовая информация, представляемая на русском, английском языках и языке разметки веб-документов HTML. Параметры, по которым проводился сравнительный анализ, время компрессии и коэффициент сжатия.

В качестве основных алгоритмов выбраны: алгоритм Хаффмена, алгоритм Лемпеля – Зива – Велча (LZW), алгоритм Шеннона-Фано, алгоритм кодирования длин серий (run-length encoding, RLE), алгоритм Лемпеля-Зива-77 (LZ77).

Суть алгоритма Хаффмена состоит в построении кодов переменной длины по вероятности вхождения символов в заданное сообщение.

Алгоритм Лемпеля – Зива – Вельча основан на создании в динамике таблицы преобразования строк: последовательно просматривая кодируемый текст,

определенным последовательностям символов (слов) ставятся в соответствие группы бит фиксированной длины (обычно 12-битные).

Кодирование Шеннона – Фано использует избыточность сообщения, состоящую в неоднородном распределении частот символов его (первичного) алфавита, заменяя коды часто встречающихся символов короткими двоичными последовательностями, а коды редко встречающихся символов – длинными последовательностями.

Алгоритм RLE (Run – Length encoding) основан на замене повторяющихся символов одним символом с указанием количества повторов.

Построение кода по алгоритму Лемпеля – Зива – 77 использует два базовых компонента: принцип скользящего окна, учитывающего ранее встречающуюся информацию, заменяя последующие вхождения ссылками на их первое вхождение, и механизм кодирования совпадений, задавая длину совпадения и смещение.

Эксперименты и оценки

Для реализации обозначенных алгоритмов был выбран язык Pyton, позволяющий за небольшое время разработать компактное приложение, обладающее кроссплат-форменностью. В качестве операционной системы выбрана широкоиспользуемая OS Windows 7. Главная задача разработанного приложения состоит в получении оценок работы алгоритмов на трех типах файлов по времени выполнения и коэффициенту сжатия для текстов различных объемов. Полученные оценки определяют целесообразность их применения при реальной компрессии текстов. Соответствующие графики даны в приведенных ниже рисунках.

Структура рисунков 1 и 2: на оси абсцисс находятся имена алгоритмов, на оси ординат – время работы алгоритма в миллисекундах. На рисунках 3 и 4 на оси ординат обозначен коэффициент компрессии, достигаемый конкретным алгоритмом. Объем исследуемого файла дан над графиком. Следует отметить, что время чтения файла, построение символьной последовательности и записи результата в выходной файл включено в общую оценку алгоритма.

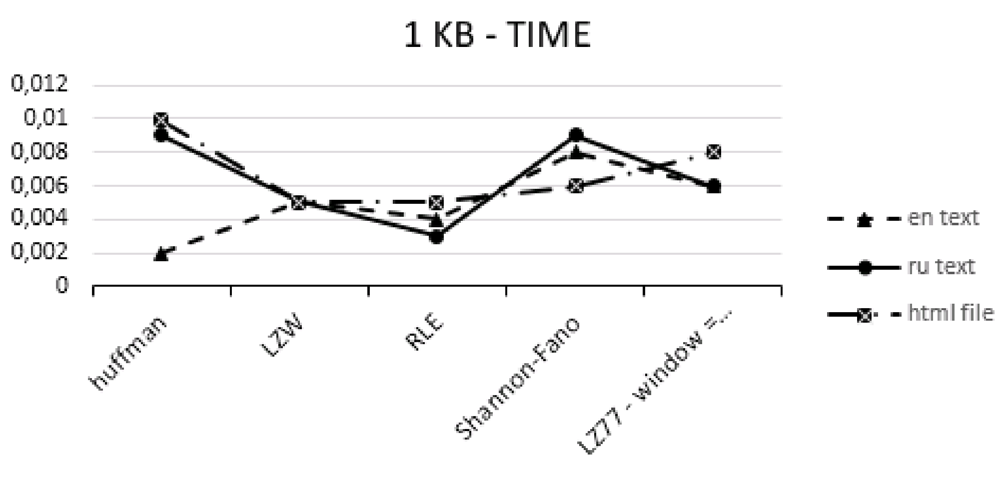


Рис. 1. График зависимости времени выполнения от объема данных (1kb)

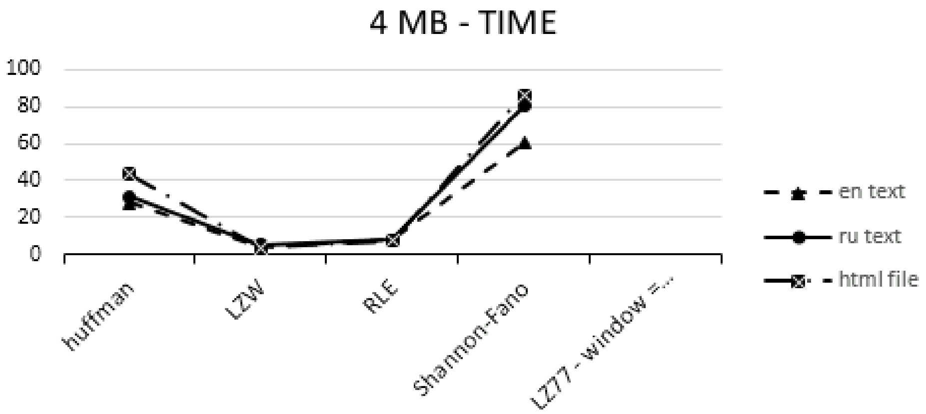


Рис. 2. График зависимости времени выполнения от объема данных (4 Mb)

На малых объемах исходных текстов видно, что алгоритм Лемпеля-Зива-Вельча ведет себя практически одинаково по времени для всех типов текстовых файлов, в то время, как алгоритм RLE показывает худшее время в среднем, а алгоритмы Хаффмена и Шеннона-Фано реагируют заметно на тип обрабатываемого файла. На больших объемах файлов наиболее заметна зависимость времени работы алгоритмов от выбранного типа файла.

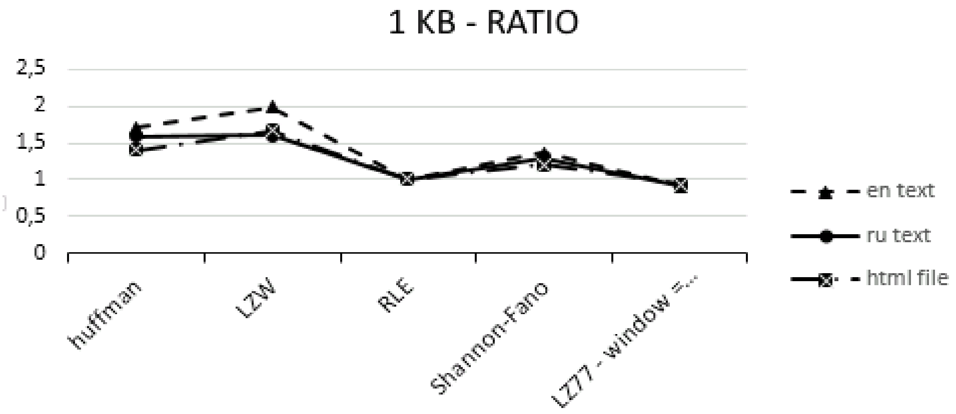


Рис. 3. График зависимости степени сжатия от объема данных (1kb)

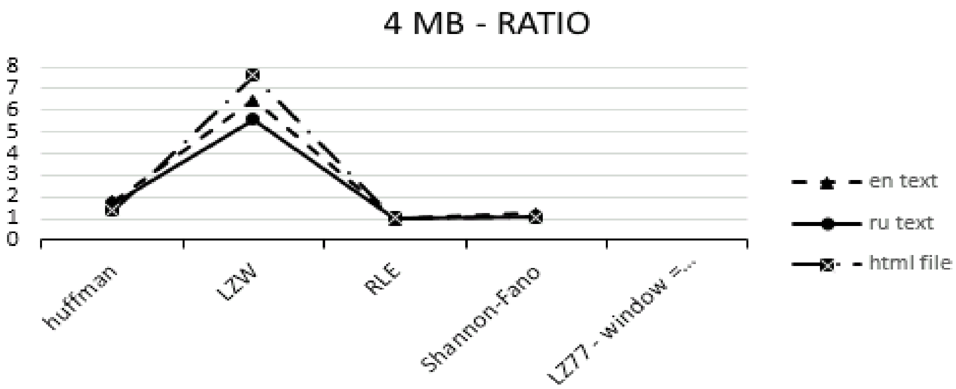


Рис. 4. График зависимости степени сжатия от объема данных (4 Mb)

По степени сжатия (рисунки 3 и 4) ряд рассматриваемых алгоритмов не зависит от рассматриваемых типов файлов при их небольших объемах. При возрастании объемов файлов для всех их типов лучшим по коэффициенту компрессии оказался алгоритм LZW.

Генерация порядка следования

Естественно, возникает вопрос, а что будет, если мы, после применения одного алгоритма для текста, попытаемся применить второй алгоритм уже для сжатого текста. На рисунке 5 приведен пример такого механизма улучшения коэффициента сжатия и сокращения времени работы. Из графика видно, что наихудшая связка по ratio Хаффмен → Шеннон-Фано. Наибольшее время дает связка Хаффмен → LZ77, но коэффициент сжатия вполне приемлем.



Рис. 5. График для комбинации алгоритмов с алгоритмом Хаффмена

Рассмотрение различных упорядочений исследуемых алгоритмов показало, что наилучшая степень сжатия достигнута при применении алгоритма Хаффмена после алгоритма LZW, и наилучшее время достигнуто при применении алгоритма Хаффмена после RLE. Файлы анализировались от 1KB до 4 MB. Для текстовых файлов это вполне достаточный интервал объемов, который позволяет сделать определенные выводы о качестве алгоритмов и эффективности механизма их упорядоченного использования.

Разработан простой графический интерфейс, позволяющий выбрать необходимый файл (кнопка «Open file»), необходимый алгоритм (по названиям) и запустить его

работу (кнопка «Count»). Результаты работы приложения (время выполнения и коэффициент сжатия) отобразятся в поле вывода.

Интерфейс приложения представлен на рис.6

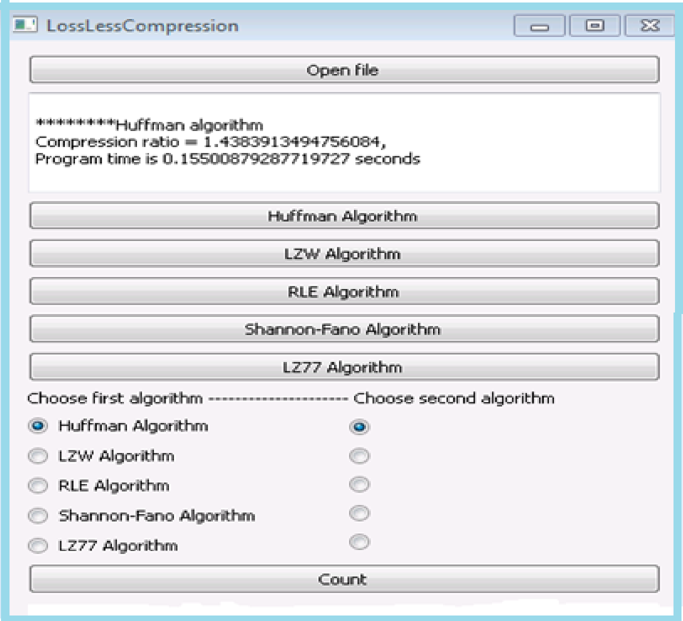


Рисунок 6 - Интерфейс приложения

Заключение

Исследования показали, что эффективность применения алгоритмов сжатия к разнотипным текстам как по коэффициенту компрессии, так и по времени работы существенно зависит от структуры текста и формы его представления. Особый интерес представляет механизм упорядоченного воздействия алгоритмов на исходный файл. Важно, что такой механизм значительно повышает коэффициент сжатия, а его эффективность существенно зависит от порядка их следования и типа файла.

Литература

1. Найдич, А. Big Data: проблема, технология, рынок/ А.Найдич// Компьютер пресс [Электронный ресурс]/ 1999-2016. – Режим доступа http://compress.ru/article.
2. Буза, М.К. Архитектура компьютеров: учебник/ М.К. Буза.-Минск: Вышэйшая школа, 2015. - 414С.
3. Сэломон, Д. Сжатие данных, изображений и звука / Д. Сэломон. – М.: Техносфера, 2004. – 368 С.
4. Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео / Д. Ватолин [и др.] под общ. Ред. Д. Ватолин – М.: Диалог – МИФИ, 2003. – 384 С.

Literatura

1. Najdich, A. Big Data: problema, tehnologija, rynok/ A.Najdich// Komp'juter press [Jelektronnyj resurs]/ 1999-2016. – Rezhim dostupa http://compress.ru/article.
2. Bouza, M.K. Arhitektura komp'juterov: uchebnik/ M.K. Bouza.-Minsk: Vyshjejshaja shkola, 2015. - 414S.
3. Sjelomon, D. Szhatie dannyh, izobrazhenij i zvuka / D. Sjelomon. – M.: Tehnosfera, 2004. – 368 S.
4. Metody szhatija dannyh. Ustrojstvo arhivatorov, szhatie izobrazhenij i video / D. Vatolin [i dr.] pod obshh. Red. D. Vatolin – M.: Dialog – MIFI, 2003. – 384 S.

RESUME

M.K. Bouza

The mechanism of increasing the efficiency of data compression

Today society generates vast volumes of information. The paper solves the problem of efficient storage of large data. For this purpose the selected algorithms of reversible data compression. We investigate the possibility of different compression algorithms: Huffmen, Lempel-Ziv-Welch, Shannon-Fano, Run-Length coding, Lempel-Ziv-77. Texts in Russian, English and HTML-files were selected as input. The volumes of input data were changed from 1 Kb to 4 Mb. For text files this volume is sufficient for research and for plausible conclusions. Two criteria of efficiency were selected: the compression ratio and compression time. Estimates of the efficacy of these algorithms were given for selected types of texts.

The paper gives the mixed mechanism of compression of text files and its effectiveness by the same criteria. It is noted that the effectiveness of the algorithms and the proposed mechanism depends on the structure of the text and the form of its presentation.

Special attention deserves the impact of the mechanism of generation of order of compression on the processed text. The values of all criteria depend on the order of the algorithms. Studies have shown that the best compression is achieved using Huffmen algorithm after the algorithm LZW. The best time of compression obtained by applying the Huffmen algorithm after the algorithm Run-Length coding . It is noted, that the worst of the bunch at the ratio Huffmen – Shannon – Fano. The greatest time compression gives a bunch of Huffmen → LZ77, while the compression ratio is quite acceptable.

The proposed mechanism of generation of the application of compression algorithms is universal. It can be used for compression of different types of data: arbitrary texts, graphics, sound, and other.

Надійшла до редакції 09.11.2016